UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/553,157	12/04/2006	Paul Frank Radford	941292-299953	1232

24239 7590 08/02/2011

MOORE & VAN ALLEN PLLC
P.O. BOX 13706
Research Triangle Park, NC 27709

EXAMINER	
VU, TUAN A	

ART UNIT	PAPER NUMBER
2193	

MAIL DATE	DELIVERY MODE
08/02/2011	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)
	10/553,157	RADFORD ET AL.
	Examiner	Art Unit
	TUAN VU	2193

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *04 December 2006*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-28* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-28* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *10 October 2005* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☒ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☒ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)	
1) ☒ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *10/10/05*.	6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to the application filed 12/04/2006

Claims 1-28 have been submitted for examination.

Claim Rejections - 35 USC § 103

2. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 1-9, 11-25, 27-28 are rejected under 35 U.S.C. 103(a) as being unpatentable over Eiche et al, USPN: 6,715,130 (herein Eiche), in view of Eder, USPubN: 20040225629 (herein Eder) and Piggot, USPN: 6,810,392 (herein Piggot), further in view of Meli et al, "Function Point Estimation Methods: A Comparative Overview", The European Software Measurement Conference, FESMA 98, 1999), pp. 1-14 (herein Meli)

As per claim 1, Eiche discloses a method for assessing a functional size of a software application or project including the steps of:

analyzing a software requirements specification and determining zero or more keywords for each requirement of the specification (e.g. Fig. 1; keywords – Fig. 3);

using a computer to cross-reference the keywords with a database stored in a computer file according to the hierarchical weight or complexity level for the keywords (Fig. 5; col. 7 lines 26-42; complexity – col. 7 lines 4-12), and

further using the computer to associate each keyword with an entry in the database (Fig. 5 and related text; col. 5 lines 56-67); using the computer to associate each keyword with an entry of the DB, thus deducing (e.g. col. 5 lines 56-67; weight 514, table 530 – Fig. 5) number of function points (e.g. *functionality value ... corresponding function points* - col. 7 lines 18-25; *function point count, total estimated function points* - col. 9 lines 39-59) related to functional size of the project (e.g. *indicators of the amount of functionality, number of features, quantity amount of functionality, assigned different weights or values ... map closely to ... function point analysis* – col 6 lines 1-28) and

combining the function points (col. 7 lines 18-25) to obtain a functional size of the software application or project (e.g. *comparable to a function point analysis on a mature SRS, quantitative assessment* – col. 6 lines 22-56; *assigned different weights or values ... map closely to ... function point analysis* – col 6 lines 1-26)

Eiche does not explicitly disclose cross-referencing the keywords with a database in terms of cross-referencing a *lexicon* stored in a computer file, nor does Eiche disclose associating each keyword with a DB entry in terms of associating each keyword *with an entry in the lexicon.*

Eiche discloses classifying to generate hierarchy of key words where weights and values (e.g. Fig. 2, 4, 5) are depicted as quantitative metrics (e.g. complexity of a domain – col. 7 lines 4-12) or characteristics like development time or function type, and to provide sufficient information to support performance analysis similar to *function point analysis* (col. 6 lines 19-56) of a SRS; and wherein certain keywords are reminiscent of the *External Interface Files* of the standard *function points* (col. 6 lines 22-24); wherein database tables supporting those hierarchies thus organized are implemented and searched as to return a weight value depending

on a domain of interest (Eiche: col. 7 lines 1-47), hence a persisted structure where each entry depict a requirement weights leading to a quantitative measure of the software to develop, each entry matching an element similar to a function type having its own level of gravity, referred as "weight". Analogous to using a database to help project a current project back in order to readjust previously stored requirement values as in Eiche's refinement approach (Eiche: col. 8 line 53 to col. 9 line 15), **Eder** discloses a multi-domain knowledge base and hierarchy levels of domain elements thereof in a framework to support analysis of metrics, establishing of contextual and environment resources for development of domain applications and to meet user's requirement (Eder: Table 6 →Table 9 pg. 3-4), where layer of such domain characteristics (including performance and function measure) after integration instances are persisted as contextbase (Eder: Figs. 6-7) having layer of entities with lexicon type of nomenclature (naming in natural language) associated with each (Eder: lexicon layer – para 0025-0032, pg. 4-5) for layer of context types help facilitate analysis and decisions as entity requirements dictate (Eder: para 0033). Using lexicon similar to the database of weights and values as in Eiche and Eder's contextbase with layered lexicons to provide natural language nomenclature support for past integration instances so as to facilitate development analysis and decision as set forth above, **Piggot** associates processing of a requirement document with consulting natural effect of parsed terms found in a lexicon (Piggot: Requirement Document, lexicon – Fig. 3) to identify candidate function points, the lexicon being extensible to augment word sense (col. 11 line 61 to col. 12 line 38) and as such a word can potentially indicate classes of function points. It would have been obvious for one of ordinary skill in the art to implement the database in Eiche so that this provide lexicon type of knowledge base organized in layers corresponding to the complexity

aspect of keywords, the lexicon for use in cross-reference a keyword based on the organized weight as in Eiche or lexicon layers as in Eder; because using effect of a lexicon in association with a keyword extracted from a natural language document (e.g. a SRS as in Piggot) would address the natural language aspect of keyword derived from Specifications document as shown in Eiche or in hierarchy knowledge and naming lexicon as in Eder, the lexicon with possibility to cross reference one or more variants of the nomenclature in view of language in which the keyword is written (using extension as in Piggot), the layered aspect of the lexicon allowing cross referencing or matching a given complexity aspect or or requirements weight as in Eiche, or having variants of nomenclature indicative of one or more function points (as in Piggot), all of which being recorded, as in Eiche or Eder, for further development reuse (e.g. to retrieve weight from a database as in Eiche from above).

Nor does Eiche explicitly disclose lexicon as also including a *function type and complexity for each keyword, and associating a keyword with entry in the lexicon, thus obtaining a function type and complexity for each keyword;* nor does Eiche explicitly disclose *using sizing standard to deduce function points for function type and complexity of each keyword.*

Eiche mentions the well-known approach for software sizing in using IFPUG (col. 2 lines 23-61) defining 5 function types; and states a high correlation in Eiche's approach with such function point analysis on quantitative estimates (col. 9 lines 33-37; col. 10 lines 4-15), where the automated estimation of development-related metric of proposed product to be carried out to meet requirements, as disclosed, are closely and linearly correlated to function point methodology (e.g. Fig. 7). Further, Eiche teaches database having distribution of key word corresponding weight, each having a connotation as low or a high per its current hierarchical

layer (col. 9 lines 1-13), hence an importance degree of "low", "high" or "average" is recognized.

Using standard function point methodology to generate metrics toward software sizing is disclosed in Piggot; that is, deducing function points for a word being cross-reference with an initially core but expendable lexicon is disclosed (Piggot: col. 12 lines 14-45) or using standard protocol of sizing as IFPUG in Piggot overview also shows quantifying or sizing of software functionality; e.g. quantitative assessment presented to a developer (col. 6 lines 5-64) prior to implementation consideration, the function types defined by that standard as EI, EO, external inquiries, ILF, and EIF, which is analogous to Eiche mention from above.

Analogous to function type mentioned in conjunction with keywords in Eiche, **Meili** discloses a sizing approach similar to Piggot and discloses the well-known 5 function types and respective complexity classification depicted as complexity weights (sec 1: Table 1, pg. 1; sec 4.2.3 pg. 9) and this would match a hierarchy of weighted elements being considered in Eiche's weighted hierarchy set forth above, where Meili mentions that, prior to adjusting to obtain the final functions points via different techniques, there is need to cataloguing of typical elements or software characteristics related to a project, each having their corresponding Unadjusted Function Point also listed (see Meili: UFP, IFPUG - sec 4.2.1; sec 4.2.3, 4.2.4 - pg. 7-10), the catalogue concept being analogous to the referencing a database or recording of weighed value in Eiche.

Based on the fact that keyword being indicator of one of the classical function points (Eiche: col. 6 lines 19-24), it would have been obvious for one of ordinary skill in the art to implement the weighted key word correspondence with the hierarchized DB values in Eiche

approach for automated of readjusting of previous values with regard to a new project consideration, so that the estimates and correlation generated thereby would also uses standard software sizing approach such as IFPUG(i.e. including defined 5 function types –as taught in Piggot) in order to match weight or complexity (Low, average, High) corresponding to each function type as shown in Meili, the persisted database or *lexicon* likewise provided with correlation of keyword with pertinent weighted values in terms of functions type and complexity values thereof (*) as shown in Meili. One would be motivated to do so because as pointed out by Eiche, the tight relationship of IFPUG and the automated readjusting approach using a persisted DB (lexicon) of hierarchized weights (i.e. complexity level values for a function type) for backward referencing would support a variety of domain applications, and such, without undue human intervention, while not far detracting from the protocol definitions and established standards in this well-known methodology; e.g. enable fast matching complexity levels to each of the IFPUG function types using this standard sizing methodology in conjunction with automated approach by which Eiche readjust values for each standardized function types using the hierarchy set up mentioned above (see *), to alleviate constant human monitoring.

As per claims 2-3, Eiche discloses method of claim 1 wherein: the step of analyzing the requirements specification includes parsing each requirement to at least isolate lexical elements and determining at least one keyword corresponding to each lexical element (e.g. Fig. 1, 3-4); and the step of cross-referencing a keyword with the lexicon includes searching the lexicon for an entry which matches said keyword (refer to the rationale in claim 1, addressing a lexicon having entry); wherein said lexical elements are individual words of a requirement (Fig. 1, 3-4).

As per claim 4, Eiche does not explicitly disclose wherein determining keywords *for a requirement* includes identifying words in the requirement (see Eiche: Fig. 1,3-4) that appear in the lexicon; but matching a keyword (in the requirement) with its weighted value that appears in a lexicon as opposed to Eiche's database has been addressed as obvious in claim 1.

As per claim 5, Eiche discloses (refer to rationale of claim 1 regarding *sizing standard*) wherein the sizing standard is a standard maintained by the International Function Point Users Group (IFPUG).

As per claim 6, Eiche does not explicitly disclose wherein: the function type associated with a keyword in the lexicon is one of *Internal Logical File, External Interface File, External Input, External Output or External Enquiry*; and the complexity associated with each keyword is one of Low Complexity, Average Complexity or High Complexity.

But the well-known definitions by IFPUG for function types (see Piggot, Eiche, Meili) in light of the correspondence thereof with complexity values based on Eiche's approach where the keywords are being classified in Eiche's database for generating of readjusted metrics supporting quantitative assessment have been mentioned in the rationale of obviousness addressing function type correlation with complexity for each keyword in the rationale of claim 1; hence the above would have been obvious for the same reasons set forth in claim 1.

As per claim 7, Eiche discloses (e.g. *functionality value ... corresponding function points* - col. 7 lines 18-25; *function point count, total estimated function points* - col. 9 lines 39-59) wherein the step of combining the function points includes summing the function points associated with all identified keywords to obtain a functional size (e.g. *indicators of the amount of functionality, number of features, quantity amount of functionality, assigned different weights*

or values ... map closely to ... function point analysis – col 6 lines 1-28) that is equal to the total number of function points.

As per claims 8-9, Eiche discloses deducing from the functional size at least one parameter associated with management of the development of the software application; wherein said at least one parameter is one or more of cost, effort, and/or development time (e.g. development time – col. 6 lines 26-33; metric … planning, costing – col. 10 lines 4-15; effort – col. 9 lines 33-38; Fig. 7).

As per claim 11, Eiche discloses a computer implemented system for assessing a functional size of a software application or project, wherein the system receives a software requirements specification as input (Fig. 1) and includes computer instruction code for:

analyzing the software requirements specification to associate each requirement with zero or more requirement keywords (refer to claim 1);

cross-referencing the requirement keywords with the database to obtain a corresponding function type (refer to claim 1) by matching requirement keywords with database keywords (Fig. 5 and text);

computing a number of function points associated with each function type (refer to claim 1)

combining the function points to obtain a functional size of the software application or project (refer to claim 1).

Eiche does not explicitly disclose a lexicon stored in a computer file, the lexicon including keywords and a function type and complexity for each keyword. But the lexicon

concept having function type and corresponding weighted complexity values in view of the database of Eiche has been rendered obvious as set forth in claim 1.

Nor does Eiche explicitly disclose cross referencing keywords with the lexicon to obtain corresponding function type and complexity from the lexicon by matching requirements keywords with lexicon keywords. This has been addressed in claim 1.

Nor does Eiche explicitly disclose computing number of FP associated with function type and complexity using the rules of a sizing standard. This also has been addressed in claim 1.

As per claims 12-14, Eiche discloses wherein the software requirements specification is received *as input from a file stored on a non-volatile storage medium of a personal computer* (document 110 – Fig. 1; col. 5 lines 1-11) but does not explicitly disclose

wherein the software requirements specification is received as input *by a user performing a cut and paste operation* from a source application; wherein the system is connected to a communications network and the *software requirements specification is received as input from a remote source over the network.*

Eiche discloses requirement document in electronic form and *MS word* format (col. 4 line 56 to col. 5 line 10); hence editing a word document using cut and paste commands would be integral to such editing tool; further receiving electronic file from commercial/communication means like internet or file transfer would entails connection to a remote source provider, thus, the "cut and paste" and receiving electronic file from a network would be deemed **either disclosed or obvious**. That is, it would have been obvious for one of ordinary skill in the art to implement the software requirements in Eiche so that this can be a form being received from a network source or editable via the cut/paste means of MS office methodology based on the above

possibility, because this would provide extensibility to the nature or format in which the SW requirement specification can be obtained or generated, using the existing methodologies at hand.

As per claim 15, Eiche does not explicitly disclose including a lexicon editor for enabling a user to modify the computer file in which the lexicon is stored, such that the lexicon keywords and corresponding function type and complexity can be changed, and new lexicon keywords and their corresponding function type and complexity can be added.

But the lexicon limitation has been addressed in claim 1.

The use of lexicon is to enable addition per need-basis or deletion of unused terms and this is shown in Piggot extensibility aspect of the knowledge base of a lexicon (Paktus, – col. 11 line 52 to col. 12 line 18). As Eiche discloses possibility to adjusting the values stored inside the hierarchy of weights stored in the DB tables via source author/adjust approach with respect to keywords encountered (col. 8 line 23 to col. 9 line 38; col. 9 line 49 to col. 10 line 3) per SRS software requirement basis which consistent with Meili adjusting of raw function point, Eder discloses data being further added to the contextbase (Para 0111, pg. 11 L col.), the lexicon aspect of augmenting a terminology or needed data respective to the adaptation for software requirement specification entails the need to associate not only the changes to the keyword but also the function point or weighed values corresponding to each added term, which is shown in Eiche (Fig or the cataloguing in Meili (sec 4.2.4 pg. 10) or the standard modifiable aspect in Function Points use (see Piggot: col. 2 lines 51-59). In view of Eiche's backward referencing and reuse of the persisted weighted hierarchy of values (e.g. projecting back – col. 8 lines 49-67) or Meili's cataloguing and readjustment, and Piggot or Eder showing of flexibility with which a

lexicon entry can be added with extended terminology or deleted entries, it would have been obvious for one of ordinary skill in the art to implement the author/readjust approach in Eiche so that, based on the need of augmenting the associated metrics related to the scope of terminology and specifications language, an editor – as in a Paktus tool - is provided to the function point developer handling the readjustment of weighted values or corresponding function points pertinent to words derived from parsing a SRS; i.e. a lexicon editor for enabling a user to modify the *lexicon* file such that the lexicon keywords and corresponding function type and complexity (see rationale in claim 1 – for lexicon having *function type and complexity* identifiers) can be changed, and new lexicon keywords and their corresponding function type and complexity can be added. One would be motivated to do so because this would enable each user with his/her own instance of environment by which to readjust the knowledge base in function to the expansion needed or construed from parsing a SRS as in Eiche or Piggot, thus lending a independency to each user without exerting too much resources in coordinating multiple instances of development or requirement analysis implicating one or more workstations operating to support the software sizing aspect using tools like IFPUG or Paktus as set forth in the endeavors of Eiche and/or Piggot or Meili.

As per claims 16-17, refer to claims 4, 6.

As per claim 18, Eiche discloses a method for assessing a functional size of a software application or project including the steps of:

analyzing a software requirements specification and determining zero or more keywords for each requirement of the specification (refer to claim 1);

cross-referencing the keywords with a lexicon stored in a computer file (refer to rationale in claim 1), said lexicon also including a function type and complexity for each keyword(refer to rationale in claim 1);

associating each keyword with an entry in the lexicon, thus obtaining a function type and complexity for each keyword(refer to rationale in claim 1);

using a functional sizing standard to deduce a number of function points for the function type and complexity of each keyword(refer to rationale in claim 1); and

combining the function points to obtain a functional size of the software application or project(refer to rationale in claim 1).

Eiche does not explicitly disclose the above steps performed in a networked computing system including a client and a server both of which are operably connected to a communications network, the client transmitting a software requirements specification to the server over the communications network; the client receiving a functional size of the software application or project from the server over the communications network. But receiving a specifications based on a network source like a service in communication with the client has been addressed as obvious in claim 14.

Nor does Eiche explicitly disclose a server operative to perform *analyzing, cross-referencing, associating, deducing, and obtaining a functional size*. Eder disclose a server in communication with clients to analyze, tracking, report-generating, prioritizing and context transforming, reevaluation of rules and perform forecasts as well as equipped with software to map context to proposed changes (see Eder: para 0110, 0112, 0115, 0118, 0119, 0121, 0122). Based on the fact that specifications data can be edited by a developer in view of the network

paradigm as set forth in claim 14, it would have been obvious for one of ordinary skill in the art to implement the software sizing in Eiche so that a developer can receive specifications file from another client in response to a request to perform the sizing steps as set forth above, the receiving developer operating as a service to implement analyzing, cross-referencing, associating, deducing, and obtaining a functional size because server functionality can include storage or database access capabilities and/or software support as shown in Eder, in a way more adequate than a client station could afford, especially in view of the security aspect related to controlling data communicated to access knowledge base as in Eder, thereby enabling intended software sizing request to be fulfilled without burdening a specific client machine.

As per claim 19, Eiche discloses a computer program product for assessing a functional size of a software application or project including computer instruction code embodied in a computer readable medium for:

analyzing a software requirements specification and determining zero or more keywords for each requirement of the specification;

cross-referencing the keywords with a lexicon stored in a computer file, said lexicon also including a function type and complexity for each keyword;

associating each keyword with an entry in the lexicon, thus obtaining a function type and complexity for each keyword;

using a functional sizing standard to deduce a number of function points for the function type and complexity of each keyword; and

combining the function points to obtain a functional size of the software application or project;

all of which having been addressed in claim 1, using the corresponding rationale.

As per claims 20-25, refer to claims 2-4, 7-9 respectively.

As per claim 27, Eiche does not explicitly disclose wherein the software requirements specification is received as input using a virtual clipboard of a computer executing the computer instruction code by a user performing a cut and paste operation from a source application. Based on the existence of a clipboard in Microsoft Office for facilitating temporary place for data being edited, this above limitation would have been obvious for the same reasoning using Microsoft Word well-known methodology as set forth in claim 12.

As per claim 28, Eiche does not explicitly disclose providing a lexicon editor that enables a user to modify the computer file in which the lexicon is stored, such that the lexicon keywords and corresponding function type and complexity can be changed, and new lexicon keywords and their corresponding function type and complexity can be added.

But this has been addressed in claim 15.

4. Claims 10, 26 are rejected under 35 U.S.C. 103(a) as being unpatentable over Eiche et al, USPN: 6,715,130, in view of Eder, USPubN: 20040225629 and Piggot, USPN: 6,810,392 (herein Piggot), and Meli et al, "Function Point Estimation Methods: A Comparative Overview", The European Software Measurement Conference, FESMA 98, 1999), pp. 1-14, further in view of Flanagan, USPubN: 20020131313 (herein Flanagan)

As per claim 10, Eiche does not explicitly disclose step of deducing the at least one parameter includes multiplying the functional size by a number representing a corresponding productivity rate. Result from producing lines of code (i.e. functional size) are measured in terms of productivity and this is disclosed in Flanagan (para 0026, pg. 3). Based on the concept

that rate at which an amount of source lines can generated, the time factor related to that would be indicative that the better the productivity, the better the production, the lesser the time; e.g. development time being inversely proportional to productivity rate. Based on the number of lines being one characteristic (Software Line Of Code) or software size requirement, the cost and effort and planning/development time as shown in Eiche (development time – col. 6 lines 26-33; cost in labor months – col. 8 lines 44-48) would also interrelate to the productivity for generating that size requirement as mentioned above; as each SW characteristic would require corresponding size or metrics under the purport in the IFPUG of Eiche where keyword weight can help deduce the above characteristics or parameter (see claims 8-9); i.e. all of which under consideration within the software sizing approach by Eiche. It would have been obvious for one of ordinary skill in the art to implement the parameters or software characteristics to contemplate such as time, effort, planning, staffing from Eiche approach (col. 10 lines 4-15; effort – col. 9 lines 33-38) so that one metric or size would be associated therewith, and in the example of parameter or characteristic such as a SLOC, this required and contemplated size can be the product using productivity rate as shown in Flanagan multiplied by the amount of time used in developing that size (a required functional size such as SLOC) because this product (multiplication-based output) would provide size indicative of the time and resources provided, as this prospected output size would establish estimates or predictive values supporting decision making relevant to using SW functional sizing as in Eiche, showing where more resources should or should not be allocated.

As per claim 26, Eiche does not explicilty disclose wherein deducing the at least one parameter includes multiplying the functional size by a number representing a corresponding productivity rate. But this has been addressed in claim 10.

Conclusion

5. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (571) 272-3735. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Lewis Bullock can be reached on (571)272-3759.

The fax phone number for the organization where this application or proceeding is assigned is (571) 273-3735 (for non-official correspondence - please consult Examiner before using) or 571-273-8300 (for official correspondence) or redirected to customer service at 571-272-3609.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100.
Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

/Tuan A Vu/

Primary Examiner, Art Unit 2193

July 31, 2011